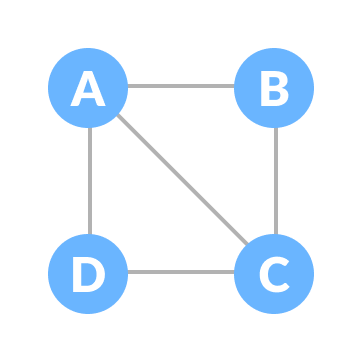
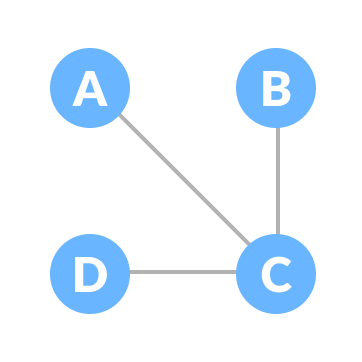
# **Spanning Tree**

Before we learn about spanning trees, we need to understand two graphs: undirected graphs and connected graphs.

An undirected graph is a graph in which the edges do not point in any direction (ie. the edges are bidirectional).

Undirected Graph

A connected graph is a graph in which there is always a path from a vertex to any other vertex.

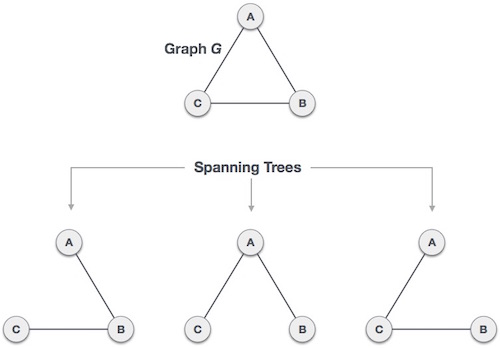
Connected Graph

## **Spanning tree**

A spanning tree is a sub-graph of an undirected connected graph, which includes all the vertices of the graph with a minimum possible number of edges. If a vertex is missed, then it is not a spanning tree. The edges may or may not have weights assigned to them.

Hence, a spanning tree does not have cycles and it cannot be disconnected.

By this definition, we can draw a conclusion that every connected and undirected Graph G has at least one spanning tree. A disconnected graph does not have any spanning tree, as it cannot be spanned to all its vertices.



We found three spanning trees off one complete graph. A complete undirected graph can have a maximum nn-2 number of spanning trees, where n is the number of nodes. In the above-addressed example, n is 3, hence 33−2 = 3 spanning trees are possible.

## General Properties of Spanning Tree

We now understand that one graph can have more than one spanning tree. Following are a few properties of the spanning tree connected to graph G −

* A connected graph G can have more than one spanning tree.
* All possible spanning trees of graph G, have the same number of edges and vertices.
* The spanning-tree does not have any cycle (loops).
* Removing one edge from the spanning tree will make the graph disconnected, i.e. the spanning tree is minimally connected.
* Adding one edge to the spanning tree will create a circuit or loop, i.e. the spanning tree is maximally acyclic.

## Mathematical Properties of Spanning Tree

* The spanning tree has n-1 edges, where n is the number of nodes (vertices).
* From a complete graph, by removing maximum e - n + 1 edges, we can construct a spanning tree.
* A complete graph can have a maximum nn-2 number of spanning trees.

Thus, we can conclude that spanning trees are a subset of connected Graph G and disconnected graphs do not have spanning tree.

## Application of Spanning Tree

A spanning tree is basically used to find a minimum path to connect all nodes in a graph. The common application of spanning trees are −

* Civil Network Planning
* Computer Network Routing Protocol
* Cluster Analysis

Let us understand this through a small example. Consider, city network as a huge graph and now plans to deploy telephone lines in such a way that in minimum lines we can connect to all city nodes. This is where the spanning tree comes into the picture.

## Minimum Spanning Tree (MST)

In a weighted graph, a minimum spanning tree is a spanning tree that has a minimum weight than all other spanning trees of the same graph. In real-world situations, this weight can be measured as distance, congestion, traffic load, or any arbitrary value denoted to the edges.

## Minimum Spanning-Tree Algorithm

We shall learn about the two most important spanning tree algorithms here −

* [Kruskal's Algorithm](https://www.tutorialspoint.com/data_structures_algorithms/kruskals_spanning_tree_algorithm.htm)
* [Prim's Algorithm](https://www.tutorialspoint.com/data_structures_algorithms/prims_spanning_tree_algorithm.htm)

## **Spanning Tree Applications**

* Computer Network Routing Protocol
* Cluster Analysis
* Civil Network Planning

## **Minimum Spanning tree Applications**

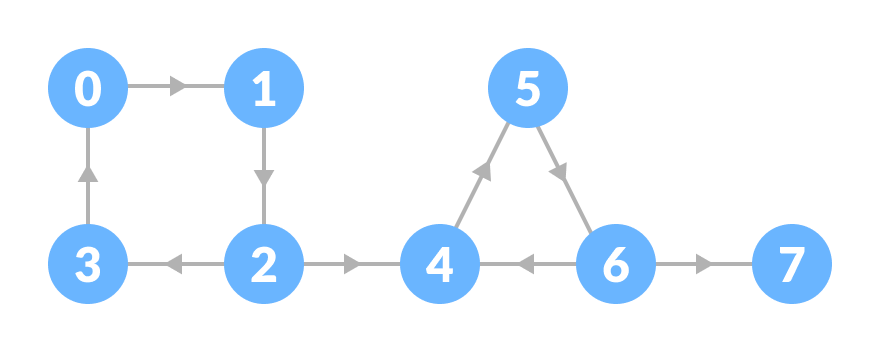
* To find paths in the map
* To design networks like telecommunication networks, water supply networks, and electrical grids.

# **Strongly Connected Components**

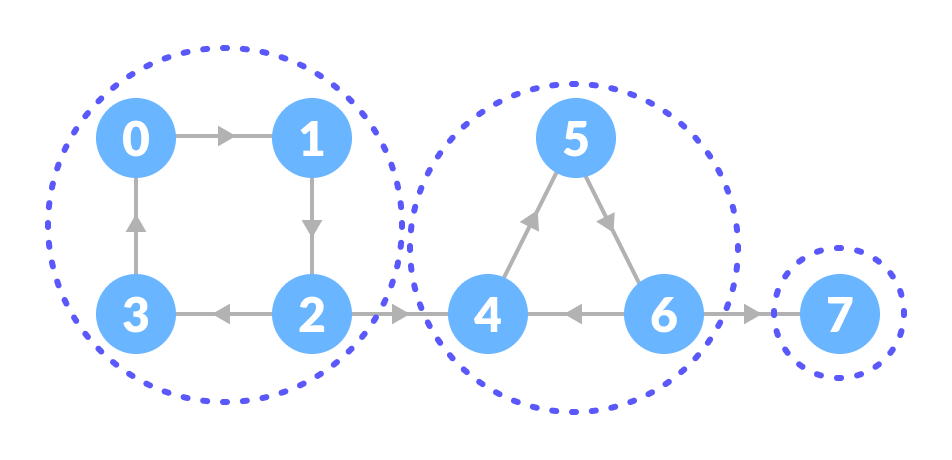
A strongly connected component is the portion of a directed graph in which there is a path from each vertex to another vertex. It is applicable only on [a directed graph](https://www.programiz.com/dsa/graph).

For example:

Let us take the graph below.

Initial graph

The strongly connected components of the above graph are:

Strongly connected components

You can observe that in the first strongly connected component, every vertex can reach the other vertex through the directed path.

These components can be found using Kosaraju's Algorithm.

## **Strongly Connected Components Applications**

* Vehicle routing applications
* Maps
* Model-checking in formal verification